REFIT: Reranker Relevance Feedback during Inference

Revanth Gangi Reddy[1] **Pradeep Dasigi**[2] **Md Arafat Sultan**[3] **Arman Cohan**[2,4]
Avirup Sil[3] **Heng Ji**[1] **Hannaneh Hajishirzi**[2,5]
[1]University of Illinois at Urbana-Champaign [2]Allen Institute for AI
[3]IBM Research AI [4]Yale University [5]University of Washington
{revanth3,hengji}@illinois.edu {pradeepd,armanc,hannah}@allenai.org
arafat.sultan@ibm.com avi@us.ibm.com

Abstract

Retrieve-and-rerank is a prevalent framework in neural information retrieval, wherein a bi-encoder network initially retrieves a pre-defined number of candidates (*e.g.*, K=100), which are then reranked by a more powerful cross-encoder model. While the reranker often yields improved candidate scores compared to the retriever, its scope is confined to only the top K retrieved candidates. As a result, the reranker cannot improve retrieval performance in terms of Recall@K. In this work, we propose to leverage the reranker to improve recall by making it provide relevance feedback to the retriever at *inference-time*. Specifically, given a test instance during inference, we distill the reranker's predictions for that instance into the retriever's query representation using a lightweight update mechanism. The aim of the distillation loss is to align the retriever's candidate scores more closely with those produced by the reranker. The algorithm then proceeds by executing a second retrieval step using the updated query vector. We empirically demonstrate that this method, applicable to various retrieve-and-rerank frameworks, substantially enhances the retrieval recall across multiple domains, languages, and modalities.

A Introduction

Information Retrieval (IR) involves retrieving a set of candidates from a large document collection given a user query. The retrieved candidates may be further reranked to bring the most relevant ones to the top, constituting a typical retrieve-and-rerank (R&R) framework [1, 2]. Reranking generally improves the ranks of relevant candidates among those retrieved, thus improving on metrics such as Mean Reciprocal Rank (MRR) [3] and Normalized Discounted Cumulative Gain (nDCG) [4], which assign better scores when relevant results are ranked higher. However, retrieval metrics like Recall@K, which mainly evaluate the presence of relevant candidates in the top K retrieved results, remain unaffected. Increasing Recall@K can be key, especially when the retrieved results are used in downstream knowledge-intensive tasks [5] such as open-domain question answering [6–8], fact-checking [9], entity linking [10–12] and dialog generation [13, 14].

Most existing neural IR methods use a dual-encoder retriever [15, 16] and a subsequent cross-encoder reranker [17]. Dual-encoder[1] models leverage separate query and passage encoders and perform a late interaction between the query and passage output representations. This enables them to perform inference at scale as passage representations can be pre-computed. Cross-encoder models, on the other hand, accept the query and the passage together as input, leaving out scope for pre-computation. The cross-encoder typically provides better ranking than the dual-encoder—thanks to its more elaborate

[1]We use the terms bi-encoder and dual-encoder interchangeably in this paper.

computation of query-passage similarity informed by cross-attention—but is limited to seeing only the retrieved candidates in an R&R framework.

Since the more sophisticated reranker often generalizes better at passage scoring than the simpler, but more efficient retriever, here we propose to use relevance feedback from the former to improve the quality of query representations for the latter directly *at inference*. Concretely, after the R&R pipeline is invoked for a test instance, we update the retriever's corresponding query vector by minimizing a distillation loss that brings its score distribution over the retrieved passages closer to that of the reranker. The new query vector is then used to retrieve documents for the second time. This process effectively teaches the retriever how to rank passages like the reranker—a stronger model—for the given test instance. Our approach, ReFIT[2], is lightweight as only the output query vectors (and no model parameters) are updated, ensuring comparable inference-time latency when incorporated into the R&R framework. Figure 47 shows a schematic diagram of our approach, which introduces a distillation and a second retrieval step into the R&R framework. By operating exclusively in the representation space—as we only update the query vectors—our framework yields a parameter-free and architecture-agnostic solution, thereby providing flexibility along important application dimensions, e.g., the language, domain, and modality of retrieval. We empirically demonstrate this effect by showing improvements in retrieval on multiple English domains, across 26 languages in multilingual and cross-lingual settings, and in different modalities such as text and video retrieval.

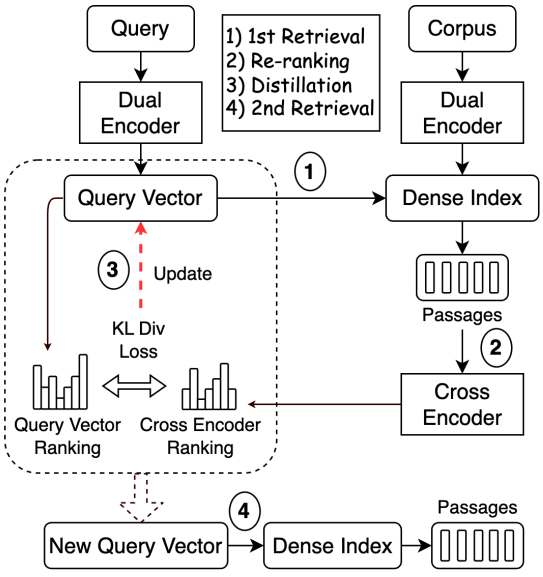

Figure 47: ReFIT: The proposed method for reranker relevance feedback. We introduce an inference-time distillation process (step 3) into the traditional retrieve-and-rerank framework (steps 1 and 2) to compute a new query vector, which improves recall when used for a second retrieval step (step 4).

Our main contributions are as follows:

- We propose ReFIT, an inference-time mechanism to improve the recall of retrieval in IR using relevance feedback from a reranker.
- Empirically, ReFIT improves retrieval performance in multi-domain, multilingual, cross-lingual and multi-modal evaluation.
- The proposed distillation step is fast, considerably increasing recall without any loss in ranking performance over a standard R&R pipeline with comparable latency.

B Related Work

Pseudo-relevance feedback: Our method has similarities with Pseudo-Relevance Feedback (PRF) [18–20] in IR: [21, 22] use the retrieved documents to improve sparse approaches via query expansion or query term reweighting, [23, 24] score similarity between a target document and a top-ranked feedback document, while [25] train a separate query encoder that computes a new query embedding using the retrieved documents as additional input. In contrast, our approach does not require customized training

[2] ReFIT stands for **Re**ranker **F**eedback at **I**nference **T**ime

feedback models or availability of explicit feedback data, as we improve the query vector by directly distilling from the reranker's output within an R&R framework.

Further, previous approaches to PRF have been dependent on the choice of retriever architecture and language; [25]'s PRF model is tied to the retriever used, [26] explore cross-lingual relevance feedback, but require feedback documents in target language and thereby could only apply to three languages, while [27] explore interpolating relevance feedback between dense and sparse approaches. On the other hand, our approach is independent of the choice of the retriever and reranker architecture, and can be used for neural retrieval in any domain, language or modality.

Distillation in Neural IR: Existing approaches primarily leverage reranker feedback *during training* of the dual-encoder retriever, to sample better negatives [28], for standard knowledge distillation of the cross-attention scores [29], to train smaller and more efficient rankers by distilling larger models [30], or to align the geometry of dual-encoder embeddings with that from cross-encoders [31]. Instead, we leverage distillation at inference time, updating only the query representation to replicate the cross-encoder's scores for the corresponding test instance. A key implication of this design choice is that unlike existing methods, we keep the retriever parameters unchanged, meaning ReFIT can be incorporated out-of-the-box into any neural R&R framework. In contrast, extending training-time distillation to new languages or modalities would require re-training the bi-encoder.

More recently, TouR [32] has proposed test-time optimization of query representations with two variants: $\text{TouR}_{\text{hard}}$ and $\text{TouR}_{\text{soft}}$. $\text{TouR}_{\text{hard}}$ optimizes the marginal likelihood of a small set of (pseudo) positive contexts. ReFIT shares similarities with $\text{TouR}_{\text{soft}}$, which uses the normalized scores of a cross-encoder over the retrieved results as soft labels. Crucially, TouR relies on multiple iterations of relevance feedback via distillation, where each iteration runs until the top-1 retrieval result has the highest reranker score (in $\text{TouR}_{\text{soft}}$) or is a pseudo-positive (in $\text{TouR}_{\text{hard}}$). This makes inference highly computationally expensive, as each additional iteration involves labeling top-K retrieval results with a reranker and then retrieving again. ReFIT improves efficiency over TouR by requiring only a single iteration of feedback that simply updates the query vector for longer, foregoing additional retrieval and reranking steps. More specifics on the inference process of the two methods can be found in §E.4. TouR was evaluated only on English phrase and passage retrieval tasks, while we demonstrate ReFIT's effectiveness in multidomain, multilingual and multimodal settings, with an empirical comparison with TouR in §E.4.

C Method

Here we discuss the standard retrieve-and-rerank (R&R) framework for IR (§C.1) and how our proposal fits into it (§C.2). While our approach can be applied to any R&R framework, we consider a text-based retriever and reranker for simplicity while elaborating our method. A multi-modal R&R framework is described in §E.3.

C.1 Retrieve-and-Rerank

R&R for IR consists of a first-stage retriever and a second-stage reranker. Modern neural approaches typically use a dual-encoder model as the retriever and a cross-encoder for reranking.

The Retriever: The dual-encoder retriever model is based on a Siamese neural network [33], containing separate Bert-based [34] encoders $E_Q(.)$ and $E_P(.)$ for the query and the passage, respectively. Given a query q and a passage p, a separate representation is obtained for each, such as the CLS output or a pooled

Algorithm 1: ReFIT

Input: Query q and its representation Q_q, retrieved passages P and their representations $\hat{P}$.
Output: Updated query representation $Q_{q,n}$

Initialize query vector $Q_{q,0} = Q_q$
Compute reranker distribution $D_{CE}(q, P)$ (Eq. 10) **for** *i in 0 to n* **do**
    Compute retriever distribution $D_{Q_{q,i}}(\hat{P})$ (Eq. 11)
Compute loss $\mathcal{L}$ (Eq. 12)
Update $Q_{q,i+1} = Q_{q,i} - \alpha \frac{\partial}{\partial Q_{q,i}} \mathcal{L}$

return $Q_{q,n}$

representation of the individual token outputs from $E_Q(q)$ and $E_P(p)$. The question-passage similarity $sim(q, p)$ is computed as the dot product of their corresponding representations: query/passage.

$$Q_q = Pool(E_Q(q)) \tag{6}$$

$$P_p = Pool(E_P(p)) \tag{7}$$

$$sim(q, p) = S(Q_q, P_p) = Q_q^T P_p \tag{8}$$

Since Eq. 8 is decomposable, the representations of all passages in the retrieval corpus can be pre-computed and stored in a dense index [35]. During inference, given a new query, the top K most relevant passages are retrieved from the index via approximate nearest-neighbor search.

The Reranker: The cross-encoder reranker model uses a Bert-based encoder $E_R(.)$, which takes the query q and a corresponding retrieved passage p together as input and outputs a similarity score. A feed-forward layer F is used on top of the CLS output from $E_R(.)$ to compute a single logit, which is used as the final reranker score $R(q, p)$. The top K retrieved passages are then ranked based on their corresponding reranker scores.

$$R(q, p) = F(CLS(E_R(q, p)) \tag{9}$$

C.2 Reranker Relevance Feedback

The main idea underlying our proposal is to compute an improved query representation for the retriever using feedback from the more powerful reranker. More specifically, we perform a lightweight inference-time distillation of the reranker's knowledge into a new query vector.

Given an input query q during inference, we use the following output provided by the R&R pipeline:

- Query representation Q_q from the retriever.
- Retrieved passages $P = \{p_1, p_2, ..., p_K\}$ and their representations $\hat{P} = [P_{p_1}, P_{p_1}, ..., P_{p_K}]$ from the retriever.
- The reranking scores $R(q, P) = [R(q, p_1), ..., R(q, p_K)]$.

Step (Device)	Retrieve & Rerank		ReFIT (K=100)
	K=100	K=125	
1st Retrieval (CPU)	40ms	40ms	40ms
Rerank (CPU)	1540ms	1925ms	1540ms
Rerank (GPU)	360ms	450ms	360ms
Distillation (CPU)	-	-	30ms
2nd Retrieval (CPU)	-	-	40ms
Total (CPU)	1580ms	1965ms	1650ms
Total (GPU)	400ms	490ms	470ms

Table 23: Comparison of inference times (in milliseconds) for different approaches, utilizing both CPU-only and GPU configurations (when reranking K passages).

Note that $\hat{P}$ above is directly obtained from the passage index and is not computed during inference.

The proposed reranker feedback mechanism begins with using the reranking scores $R(q, P)$ to compute a cross-encoder ranking distribution $D_{CE}(q, P)$ over passages P as follows:

$$D_{CE}(q, P) = \text{softmax}([R(q, p_1), ..., R(q, p_K)]) \tag{10}$$

The query and passage representations from the retriever are used to compute a similar distribution $D_{Q_q}(\hat{P})$ over P:

$$D_{Q_q}(\hat{P}) = \text{softmax}([Q_q^T P_{p_1}, ..., Q_q^T P_{p_K}]) \tag{11}$$

Next, we compute the loss as the KL-divergence between the retriever and reranker distributions:

$$\mathcal{L} = D_{KL}(D_{CE}(q, P)||D_{Q_q}(\hat{P})) \tag{12}$$

which is then used to update the query vector via gradient descent. The query vector update process is repeated for n times, where n is a hyper-parameter. A schematic description of the process can be found in Algorithm 1.

Finally, the updated query vector $Q_{q,n}$ is used for a second-stage retrieval from the passage index. From dual-encoder retrieval with the updated $Q_{q,n}$, we aim to achieve better recall than with the initial Q_q, while obtaining a ranking performance that is comparable with that of the reranker.

D Experimental Setup

D.1 Distillation Process

We observe that the output scores from the dual-encoder and the cross-encoder models are not bounded to specific intervals. Hence, we do min-max normalization separately on the query vector's scores $Q_q^T \hat{P}$ (from the dual-encoder) and the cross encoder's scores $R(q, P)$ to bring the two scoring distributions closer. Further, the cross-encoder tends to have peaky scoring distributions, hence we use a temperature T (= 2 after tuning) while computing the softmax $D_{CE}(q, P)$ over the cross-encoder scores. After tuning on the MS Marco dev set, we set the number of updates n=100 with learning rate α=0.005.

	BM25	ANCE	RocketQA v1	RocketQA v2	RocketQA v1+Rerank	RocketQA v1+ReFIT	Contriever	Contriever + Rerank	Contriever + ReFIT
MS MARCO	65.8	85.2	88.4	88.7	89.4	**90.0***	89.1	89.9	**90.5***
Trec-COVID	49.8	45.7	48.5	46.4	52.0	**52.9**	40.7	43.8	**51.5***
NFCorpus	25.0	23.2	26.9	25.9	27.4	**29.2***	30.0	29.5	**31.9***
NQ	76.0	83.6	91.1	89.8	91.8	**92.7***	92.5	93.3	**94.2***
HotpotQA	74.0	57.8	69.8	67.7	71.4	**73.3***	77.7	78.6	**80.4***
FiQA	53.9	58.1	63.6	61.2	**64.3**	63.8	65.6	**65.9**	65.6
DBPedia	39.8	31.9	45.7	43.4	47.6	**50.2***	54.1	56.0	**57.3***
Scidocs	35.6	26.9	31.8	29.3	33.1	**35.5***	37.8	38.3	**40.1***
FEVER	93.1	90.0	92.6	92.5	92.8	**93.7***	94.9	95.3	**95.5***
Climate-FEVER	43.6	44.5	47.4	48.7	49.3	**53.6***	57.4	59.0	**59.5**
Scifact	90.8	81.6	88.1	85.4	89.0	**89.9***	94.7	94.4	**95.2***
Average	58.9	57.1	63.1	61.7	64.4	**65.9***	66.8	67.6	**69.0***

Table 24: Recall@100 (in %) on the English BEIR benchmark. Performance of ReFIT is shown for different choices of underlying retrievers. RocketQAv2 [39] corresponds to a training-time distillation baseline. Improvements marked with * are statistically significant at $p < 0.05$ as per paired t-test.

D.2 Rerank Baseline

ReFIT introduces the additional overhead of distillation and a second retrieval step into the R&R framework. We note that distillation latency (in Algorithm 1) in linear in the number of updates n. Table 23 compares the inference latency of our method with that of standard R&R, assuming K=100 passages are to be reranked and n=100 updates are used during distillation. We highlight that our distillation process is lightweight and takes just 30ms on a CPU. We see that the additional distillation and retrieval steps increase the latency of inference by roughly 17.5% when using a GPU (or 4.4% for CPU);[3] in that same amount of time, vanilla R&R can process a total of 125 passages on the GPU (see Table 23), to potentially increase Recall@100. Hence, and for fair comparison, we evaluate against a *Rerank* baseline that is allowed to retrieve and rerank 125 passages. We note that both ReFIT and the *Rerank* baseline use the same retriever and reranker, and are evaluated on Recall@100.

D.3 Retriever and Reranker

We use Contriever [36] as the underlying retriever (unless otherwise mentioned), which has been pre-trained with an unsupervised contrastive learning objective on a large-scale collection of Wikipedia and CCNet documents. Contriever is a dual-encoder retriever that outperforms traditional term-matching methods, BM25 and recent dense approaches e.g. DPR [15] and ANCE [37]. For retrieval in both English and other languages, we use the publicly available version of Contriever, fine-tuned on MS MARCO [38]. Our English[4] and multilingual[5] rerankers are based on sentence transformers.

E Results

E.1 English Retrieval in Multiple Domains

We evaluate English retrieval performance on the BEIR benchmark [40], comprising training and in-domain test instances from MS MARCO and out-of-domain evaluation data from a number of scientific,

[3] 24-core AMD EPYC 7352 CPU and 80GB A100 GPU.
[4] cross-encoder/ms-marco-MiniLM-L-6-v2
[5] cross-encoder/mmarco-mMiniLMv2-L12-H384-v1

	mBERT	XLM-R	Contriever	Rerank	ReFIT
Arabic	81.1	79.9	88.7	89.5	**90.9***
Bengali	88.7	84.2	91.4	91.4	**95.9***
English	77.8	73.1	77.2	78.7	**81.8***
Finnish	74.2	81.6	88.1	88.9	**91.0***
Indonesian	81.0	87.4	89.8	90.5	**93.7***
Japanese	76.1	70.9	81.7	82.5	**85.2***
Korean	66.7	71.1	78.2	**81.0**	80.2
Russian	77.6	74.1	83.8	85.7	**87.3**
Swahili	74.1	73.9	91.4	**92.0**	90.5
Telugu	89.5	91.2	96.6	97.0	**97.5**
Thai	57.8	89.5	90.5	91.6	**93.3***
Average	76.8	79.7	87.0	88.1	**89.7***

Table 25: Recall@100 (in %) on the multilingual Mr.TyDi benchmark. Rerank and ReFIT use Contriever as the underlying retriever. * corresponds to statistical significance at $p < 0.05$ (paired t-test).

biomedical, financial, and Wikipedia-based retrieval datasets[6].

Firstly, we compare our inference-time distillation approach against a training-time distillation method. We use RocketQAv1 [41] as the underlying retrieval model and RocketQAv2 [42] as the retriever distilled at training time from the cross-encoder. We also compare with a *Rerank* (K=125) baseline, which improves Recall@100 by reranking the top 125 passages (retrieved by RocketQAv1). Moreover, we also demonstrate the effectiveness of ReFIT with a different underlying retrieval model, in this case, Contriever.

Table 24 shows Recall@100 results on the BEIR benchmark. Firstly, we see that ReFIT consistently outperforms all baselines. Next, RocketQAv2 shows improvement over RocketQAv1 on MS MARCO, which is the dataset used for training-time distillation of RocketQAv2. However, RocketQAv2's performance degrades on out-of-domain datasets from the BEIR benchmark. This is unsurprising, since the training-time distillation approach is limited to the bi-encoder seeing the cross-encoder's relevance labels only in the source domain, i.e. the domain used for training (MS MARCO in this case). As a result, the training-time distillation approach may not generalize well to unseen domains (BEIR in this case). In contrast, ReFIT offers the key advantage of learning from target-domain pseudo labels provided by the reranker *at inference*, which yields improved out-of-domain generalization.

E.2 Retrieval in More Languages

Multilingual Retrieval

We also evaluate on Mr. TyDi [43], a multilingual IR benchmark derived from TyDi QA [44], where given a question in one of 11 languages, the goal is to retrieve candidates from a pool of Wikipedia documents in the same language. Our underlying retriever is the multilingual version of Contriever. Other baseline retrieval models are mBERT and XLM-R [45], in addition to the *Rerank* (K=125) baseline. Table 25 shows Recall@100 for the different systems on Mr.TyDi. Here again, ReFIT yields significant improvement over all baselines on most languages.

Cross-lingual Retrieval

For our cross-lingual experiments, we used the MKQA benchmark [46]. MKQA involves retrieving passages from the English Wikipedia corpus for questions that are posed in 26 different languages.

[6]We omit some datasets due to license & versioning issues.

	avg	en	ar	fi	ja	ko	ru	es	sv	he	th	da	de	fr
mBERT	57.9	74.2	44.0	51.7	55.7	48.2	57.4	63.9	62.7	46.8	51.7	63.7	59.6	65.2
XLM-R	59.2	73.4	42.4	57.7	53.1	48.6	58.5	62.9	67.5	46.9	61.5	66.9	60.9	62.4
Contriever	65.6	75.6	53.3	66.6	60.4	55.4	64.7	70.0	70.8	59.6	63.5	72.0	66.6	70.1
Rerank	66.4	76.0	54.5	67.5	61.5	56.7	65.8	70.5	71.6	60.8	64.9	72.7	67.5	70.6
ReFIT	**68.2**	**76.6**	**58.0**	**68.8**	**64.7**	**59.3**	**68.4**	**72.5**	**73.1**	**62.9**	**66.5**	**74.1**	**70.1**	**72.5**

	it	nl	pl	pt	hu	vi	ms	km	no	tr	zh-cn	zh-hk	zh-tw
mBERT	64.1	66.7	59.0	61.9	57.5	58.6	62.8	32.9	63.2	56.0	58.4	59.3	59.3
XLM-R	58.1	66.4	61.0	62.0	60.1	62.4	66.1	**46.6**	65.9	60.6	55.8	55.5	55.7
Contriever	70.3	71.4	68.8	68.5	66.7	67.8	71.6	37.8	71.5	68.7	64.1	64.5	64.3
Rerank	70.8	72.0	69.9	69.3	67.5	68.7	72.0	38.6	72.3	69.3	65.1	65.4	65.2
ReFIT	**72.4**	**73.6**	**71.1**	**71.5**	**68.9**	**70.5**	**73.3**	39.9	**73.3**	**70.7**	**67.5**	**67.4**	**66.9**

Table 26: Recall@100 (in %) on the cross-lingual MKQA benchmark. Rerank and ReFIT use Contriever as the underlying retriever. All improvements are statistically significant at $p < 0.05$ (paired t-test).

Following [36], we discard unanswerable questions and questions with a yes/no answer or a long answer, leaving 6,619 queries per language in the final test set. Table 26 compares Recall@100 of different models on MKQA. ReFIT again outperforms, leading the nearest baseline (*Rerank*) by about 2 points on average, and with improvements on all 26 MKQA languages.

E.3 Multi-modal Retrieval

A key advantage of ReFIT is that it can operate independently of the choice of architecture for the bi-encoder and the cross-encoder, and is therefore not limited to working on only text input. To demonstrate this, we apply our method to retrieval in a multi-modal setting. Specifically, we consider text-to-video retrieval, which involves retrieving videos that are relevant to a given textual query.

The retriever and reranker for this experiment are based on BLIP [47], a state-of-the-art vision-language model that comprises two unimodal encoders and an image-grounded text encoder. The unimodal encoders encode image and text separately, akin to dual-encoders in text-to-text retrieval, and are trained with an Image-Text Contrastive (ITC) loss. The image-grounded text encoder injects visual information into the text encoder by incorporating a cross-attention layer, similar to a text-to-text cross-encoder, and is trained with an Image-Text Matching (ITM) loss. We refer the reader to [47] for a more detailed description of BLIP's architecture and the pre-training objectives. BLIP can thus be used for retrieval with the unimodal encoders (which we refer to as BLIP$_{ITC}$), and for reranking with the image-grounded text encoder (which we refer to as BLIP$_{ITM}$). We use the output from BLIP$_{ITM}$ as the reranker distribution, which is then used to compute the distillation loss for updating the query representation that is output by BLIP$_{ITC}$.

We evaluate using Recall@100 on the MSRVTT [48] text-to-video retrieval dataset, with BLIP$_{ITC}$ [47] being our primary retrieval-only baseline along with other baselines taken from [47]. The *Rerank* baseline uses BLIP$_{ITM}$ to rerank K=125 videos retrieved by BLIP$_{ITC}$. Table 27 compares performance on the 1k test split of MSRVTT. We see that BLIP$_{ITM}$ yields better ranking (as evident from higher Recall@10) than BLIP$_{ITC}$ as expected, but shows only minor gains in Recall@100. Crucially, ReFIT improves Recall@100 over the already strong BLIP$_{ITC}$ retriever, without a noticeable drop in Recall@10 compared to the BLIP$_{ITM}$ reranker.

Method	R@10	R@100
MIL-NCE	32.4	-
VideoCLIP	30.0	-
FiT	51.6	-
BLIP_{ITC}	69.0	92.1
Rerank (BLIP_{ITM})	74.6	92.3
ReFIT	**74.7**	**92.9**

Table 27: Recall of text-to-video retrieval methods on the MSRVTT benchmark.

	NQ	EntityQ	BEIR	Mr.TyDi
Retrieve	86.1	70.1	66.8	87.0
Rerank	86.8	71.2	67.6	88.1
$\text{TouR}_{\text{hard}}$	87.0^{+}	71.9^{+}	68.4	88.7
$\text{TouR}_{\text{soft}}$	87.2^{+}	72.5^{+}	68.1	88.1
ReFIT	**87.6**	**72.6**	**69.2**	**89.7**

Table 28: Recall@100 numbers for comparison of ReFIT with both variants of TouR. $^{+}$ corresponds to numbers directly taken from [32].

E.4 Comparison with TouR

In this section, we compare the performance of ReFIT with TouR on the passage retrieval benchmarks used in [32], NQ [49] and EntityQuestions [50] as well as the multidomain BEIR and multilingual Mr.TyDi benchmarks. For NQ and EntityQuestions, we use the same retriever [15] and reranker [51] as in [32]. The retriever and the reranker for BEIR and Mr.TyDi are the same as described in §D.3. In Table 28, we can see that ReFIT consistently outperforms both TouR variants across various datasets. We believe that the lower performance of TouR can be attributed to its early stopping criterion for distillation updates. Specifically, TouR performs relevance feedback for 3 iterations, wherein in each iteration, distillation into the query vector continues until the top-1 retrieval result has the highest reranker score (for $\text{TouR}_{\text{soft}}$) or is a pseudo-positive (for $\text{TouR}_{\text{hard}}$). In contrast, ReFIT makes more distillation updates but for only one iteration (we do $n = 100$ updates, which has been tuned). This makes it also considerably faster than TouR, as each additional iteration of relevance feedback in TouR comes with a high computational overhead (§B). We show in §F.4 that ReFIT can further benefit from multiple rounds of relevance feedback with continuous improvements over the course of three iterations.

E.5 ReFIT for multi-vector dense retrieval

Our experiments thus far have been focused on single-vector dense retrieval, where queries and passages are encoded as individual vectors. Multi-vector retrieval models like ColBERT [16, 52], on the other hand, compute token-level query and passage representations, subsequently employing a late-interaction mechanism for scoring. This section explores the application of ReFIT to multi-vector retrieval, specifically, ColBERTv2 [52]. In this case, distillation (Step 3 in Figure 47) updates embeddings of individual tokens in the query. We present results in Table 29 on a subset[7] of the BEIR dataset, which clearly show that ReFIT can be effectively extended to multi-vector dense retrieval, as it consistently surpasses the performance of the ColBERTv2 retriever and outperforms the Rerank baseline (with $K = 125$) in most cases. Notably, the training of ColBERTv2 [52] involved the use of a reranker's scores for supervision; our results in this section thus reinforce the finding of §E.1 that ReFIT's inference-time distillation can be superior to ordinary knowledge distillation during training.

	ColBERTv2	Rerank	ReFIT
NFCorpus	27.7	28.0	**28.8**
FiQA	62.8	63.7	**64.3**
Scidocs	35.8	36.6	**38.5**
Scifact	89.4	**90.2**	90.1

Table 29: Recall@100 (in %) on a subset of the English BEIR benchmark, with Rerank and ReFIT using ColBERTv2 as the underlying retriever.

[7] Owing to the substantially larger index size inherent to multi-vector dense retrieval, we restrict this study to subsets of BEIR with $< 100k$ passages in the retrieval corpus.

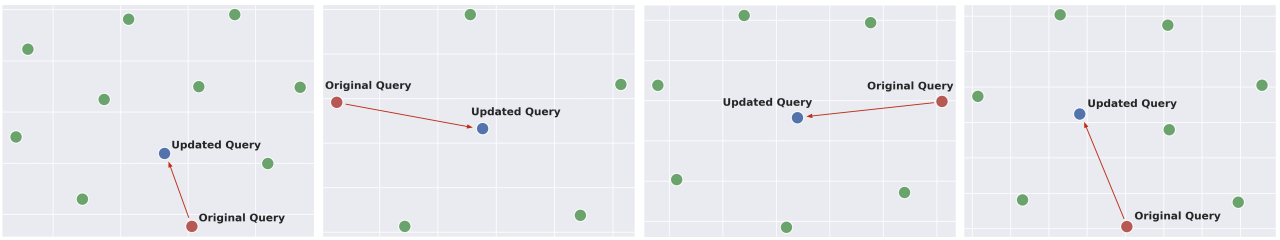

Figure 48: t-SNE plots for some examples from BEIR, with the query vectors shown alongside the corresponding positive passages. The updated query vectors after ReFIT are now closer to the positive passages (in green).

Query	Initial Retrieval (within Reranker Top-5)	Newly Retrieved Positive
treating tension headaches without medication	Most intermittent tension-type headaches are easily treated with over-the-counter medications, including: 1 Aspirin. 2 Ibuprofen (Advil, Motrin IB, others) 3 Acetaminophen (Tylenol, others)	Instead of popping a pill when you get a headache, toss some almonds. For everyday tension-type headaches, almonds can be a natural remedy and a healthier alternative to other medicine.
who drives the number 95 car in nascar	On October 2013, it was announced that McDowell would be moving to Leavine Family Racing's No. 95 Ford for the 2014 NASCAR Sprint Cup Series season. McDowell failed to qualify for the Daytona 500.	Michael Christopher McDowell is an American professional stock car racing driver. He currently competes full-time in the Monster Energy NASCAR Cup Series, driving the No. 95 Chevrolet SS for Leavine Family Racing.
who plays addison shepherd on grey's anatomy	In 2005, she was cast in her breakout role in the ABC series Grey's Anatomy, as Dr. Addison Montgomery, the estranged spouse of Derek Shepherd.	Kathleen Erin Walsh is an American actress and businesswoman. Her roles include Dr. Addison Montgomery on the ABC television dramas Grey's Anatomy and Private Practice.

Table 30: Examples of how initial retrievals highly ranked (top-5) by the reranker (middle) helps retrieve new positives (right) via the updated query vector, due to important lexical and semantic overlap (highlighted in green). The text that contains the answer to the query is shown in red.

F Discussion and Analysis

This section describes additional experiments, providing further insights into ReFIT.

F.1 Query vectors: the original and the new

To better understand how the updated query vector after reranker relevance feedback improves recall, we take a closer look at the query and passage vectors computed for a set of BEIR examples. Figure 48 shows t-SNE plots for four such examples, where each dot represents a vector, and the distance between any two points is their cosine distance. As the figure shows, the reranker feedback brings the query vector in each case closer to the corresponding positive passage vectors, making the query align with an increased number of relevant passages and consequently improving recall. Across different datasets in BEIR, we observed that the new query vector is also closer to the initially retrieved positives by 5-16%.

We observe that the new positives discovered by the updated query vector are closest to a passage in the reranker's top 5 in 26% of the cases (38% for top 10; 55% for top 20), confirming an effective transfer of the reranker's knowledge into the query vector. Table 30 provides some examples, showing how specific words and phrases in a passage within the reranker top-5 help retrieve additional candidates with lexical/semantic overlap (highlighted in green) via relevance feedback. Interestingly, in the fourth example, an incorrect passage highly scored by the reranker leads to the subsequent retrieval of an actual positive candidate.

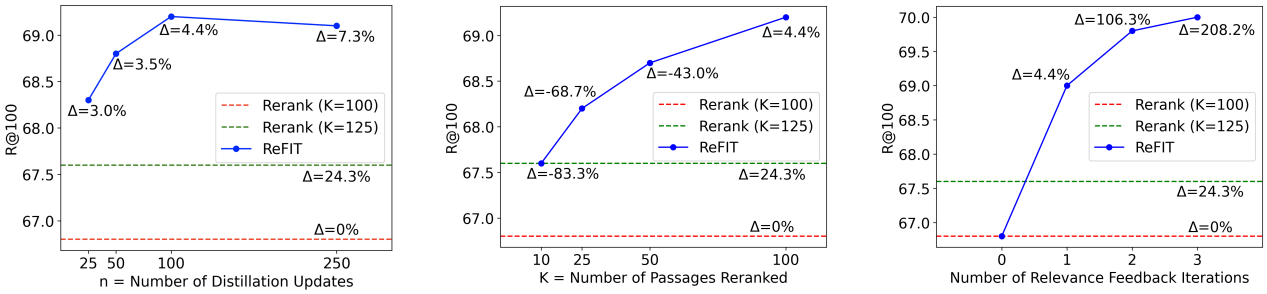

Figure 49: Variation of ReFIT performance with distillation updates n (left), reranked passages K used for distillation supervision (center) and relevance feedback iterations (right). Δ corresponds to change in latency with respect to the standard R&R framework with K=100 (CPU-only configuration).

F.2 How much additional latency does our approach introduce?

Our proposed method introduces a distillation and an additional retrieval step into the standard R&R framework. While retrieval takes constant time with respect to the number of updates n in Algorithm 1, the latency of distillation is directly proportional to n. Figure 49 (left) demonstrates the effect of varying n on both the latency and performance of our approach. The extra latency is computed with respect to a standard R&R framework that runs with K=100. With a mere 4.4% increase in latency (for when n=100), our method produces a gain that is significantly larger than a more computationally expensive reranking of K=125 candidates which in turn corresponds to 24.3% increase in latency on a CPU. Thereby, we demonstrate that, under latency constraints, our approach can be made faster by simply lowering the number of updates, while still surpassing the conventional strategy of reranking a larger pool of candidates for improving recall.

F.3 How do smaller K values affect results?

Our experiments described thus far are run in the standard setting of K=100: 100 passages are retrieved, reranked and subsequently used to distill the reranker score distribution into the new query. Here we investigate how ReFIT performs as we vary K. Smaller values of K correspond to a faster R&R pipeline (as lower number of candidates are reranked), but it comes at the expense of the target teacher distribution now providing lesser supervision. Figure 49 (center) shows Recall@100 of the post-relevance feedback retrieval step on BEIR for different values of K. While a higher K expectedly leads to a higher recall in general, we observe performance improvements over directly reranking 125 passages, even when considerably smaller number of passages are used for distillation. Our approach can thus be easily tuned to achieve different accuracy-speed trade-offs depending on the requirements of the target application.

F.4 Can multiple iterations of relevance feedback further improve results?

Our relevance feedback approach improves recall when the updated query vector is used for a second retrieval step. Here we examine if further improvements are possible from more iterations of relevance feedback, i.e., running the following operations in a loop: (1) rerank the retrieval results from the previous iteration, (2) update the query vector via distillation from the reranker distribution, and (3) retrieve again. We note that this experiment operates under the assumption of a relaxed time budget, as the computationally expensive reranker must be executed N times. Figure 49 (right) shows performance on BEIR from N iterations of relevance feedback, with $N = 0$ corresponding to baseline retrieval. We can see that recall improves with each additional round of relevance feedback; the biggest gain comes in the first round ($N = 1$) and performance saturates after $N = 2$.

F.5 Further Discussion

The curious case of zero initial positives:

In §F.1, we presented an example where our method leverages a close negative among the initially retrieved candidates to later retrieve a positive passage. We find that in 24% of the cases where the first-stage retriever retrieves no positive passages, our method can improve recall in a similar fashion. Among all cases where recall improves, however, 75% have at least one positive in the top retrieved results. These results indicate that while the presence of positive candidates in the initial retrieval is useful, our relevance feedback approach can also generally leverage informative negatives to update the query vector in the right direction.

Choice of Reranker:

In the experiments comparing our approach to the R&R framework, we used an efficient (yet high-performing) reranker both in the baseline model and as the teacher model for distillation. Would the results have been different if we used a more powerful (but computationally expensive) reranker instead? To find an answer, it is essential to note that the final recall of an R&R engine is inherently limited by the underlying retriever. For instance, the Recall@100 of an R&R pipeline with K=125 cannot exceed the Recall@125 of the underlying retriever, irrespective of the quality of the reranker. The Recall@100 of ReFIT (BEIR: 69.2, Mr. TyDi: 89.7, MKQA: 68.2 and MSRVTT: 92.9) is consistently higher than the Recall@125 of the baseline retriever (BEIR: 68.9, Mr. TyDi: 88.2, MKQA: 66.9 and MSRVTT: 92.8). These results clearly suggest that even the best reranker baseline would fail to attain the recall of our method. Further, we can expect a better reranker to improve the recall of ReFIT since leveraging a stronger teacher model for distillation should lead to a better student (retriever query vector).

G Conclusion and Future Work

We demonstrate that query representations can be improved using feedback from a cross-encoder reranker *at inference time* for better performance of dual-encoder retrieval. This work proposes for distillation using relevance feedback from the reranker as a better and faster alternative to the traditional strategy of reranking a larger pool of candidates for improving recall. ReFIT is lightweight and improves retrieval accuracy across different domains, languages and modalities over a state-of-the-art retrieve-and-rerank pipeline with comparable latency. Future work will focus on the potential integration of textual relevance feedback from large language models (LLMs). Additionally, a promising area of exploration lies in enhancing the interpretability by examining how relevance feedback influences the significance of individual query terms within the query representation.

H Limitations

ReFIT introduces an additional latency into a traditional retrieve-and-rerank framework. The distillation time is only dependent on the number of updates, and is unaffected by the model architecture and number of retrieved passages; the overall additional latency (as per Table 23) amounts to an extra 17.5% on GPU (or 4.4% on CPU) when the number of retrieved passages K=100. However, it is noteworthy that ReFIT remains faster and exhibits superior performance compared to the standard approach of reranking a larger pool of candidates for improving recall. Moreover, the efficacy of our approach is contingent upon the reranker providing a better ranking than the retriever. We anticipate that our method might provide minimal gains in situations where the retriever performs similar to the reranker.

References

[1] S. WANG, M. YU, J. JIANG, W. ZHANG, X. GUO, S. CHANG, Z. WANG, T. KLINGER, G. TESAURO, and M. CAMPBELL, "Evidence aggregation for answer re-ranking in open-domain question answering.(2018)," in Proceedings of the 6th International Conference on Learning Representation, Vancouver, Canada, 2018 April 30-May, vol. 3, pp. 1–14.

[2] M. Hu, Y. Peng, Z. Huang, and D. Li, "Retrieve, read, rerank: Towards end-to-end multi-document reading comprehension," in Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 2285–2295, 2019.

[3] N. Craswell, Mean Reciprocal Rank, pp. 1703–1703. Boston, MA: Springer US, 2009.

[4] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of ir techniques," ACM Transactions on Information Systems (TOIS), vol. 20, no. 4, pp. 422–446, 2002.

[5] F. Petroni, A. Piktus, A. Fan, P. Lewis, M. Yazdani, N. De Cao, J. Thorne, Y. Jernite, V. Karpukhin, J. Maillard, et al., "Kilt: a benchmark for knowledge intensive language tasks," in Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 2523–2544, 2021.

[6] D. Chen, A. Fisch, J. Weston, and A. Bordes, "Reading wikipedia to answer open-domain questions," in Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1870–1879, 2017.

[7] D. Chen and W.-t. Yih, "Open-domain question answering," in Proceedings of the 58th annual meeting of the association for computational linguistics: tutorial abstracts, pp. 34–37, 2020.

[8] R. Gangi Reddy, B. Iyer, M. A. Sultan, R. Zhang, A. Sil, V. Castelli, R. Florian, and S. Roukos, "Synthetic target domain supervision for open retrieval qa," in Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1793–1797, 2021.

[9] J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal, "Fever: a large-scale dataset for fact extraction and verification," in Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pp. 809–819, 2018.

[10] J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum, "Robust disambiguation of named entities in text," in Proceedings of the 2011 conference on empirical methods in natural language processing, pp. 782–792, 2011.

[11] A. Sil and A. Yates, "Re-ranking for joint named-entity recognition and linking," in Proceedings of the 22nd ACM international conference on Information & Knowledge Management, pp. 2369–2374, 2013.

[12] A. Sil, G. Kundu, R. Florian, and W. Hamza, "Neural cross-lingual entity linking," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, 2018.

[13] E. Dinan, S. Roller, K. Shuster, A. Fan, M. Auli, and J. Weston, "Wizard of wikipedia: Knowledge-powered conversational agents," in International Conference on Learning Representations, 2018.

[14] M. Komeili, K. Shuster, and J. Weston, "Internet-augmented dialogue generation," in Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 8460–8478, 2022.

[15] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih, "Dense passage retrieval for open-domain question answering," in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 6769–6781, 2020.

[16] O. Khattab and M. Zaharia, "Colbert: Efficient and effective passage search via contextualized late interaction over bert," in Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, pp. 39–48, 2020.

[17] R. Nogueira and K. Cho, "Passage re-ranking with bert," arXiv preprint arXiv:1901.04085, 2019.

[18] J. Rocchio, "Relevance feedback in information retrieval," The Smart retrieval system-experiments in automatic document processing, pp. 313–323, 1971.

[19] Y. Lv and C. Zhai, "Adaptive relevance feedback in information retrieval," in Proceedings of the 18th ACM conference on Information and knowledge management, pp. 255–264, 2009.

[20] H. Li, A. Mourad, B. Koopman, and G. Zuccon, "How does feedback signal quality impact effectiveness of pseudo relevance feedback for passage retrieval?," arXiv preprint arXiv:2205.05888, 2022.

[21] M. Bendersky, D. Metzler, and W. B. Croft, "Parameterized concept weighting in verbose queries," in Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval, pp. 605–614, 2011.

[22] J. Xu and W. B. Croft, "Quary expansion using local and global document analysis," in Acm sigir forum, vol. 51, pp. 168–175, ACM New York, NY, USA, 2017.

[23] C. Li, Y. Sun, B. He, L. Wang, K. Hui, A. Yates, L. Sun, and J. Xu, "Nprf: A neural pseudo relevance feedback framework for ad-hoc information retrieval," in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 4482–4491, 2018.

[24] Z. Zheng, K. Hui, B. He, X. Han, L. Sun, and A. Yates, "Bert-qe: Contextualized query expansion for document re-ranking," in Findings of the Association for Computational Linguistics: EMNLP 2020, pp. 4718–4728, 2020.

[25] H. Yu, C. Xiong, and J. Callan, "Improving query representations for dense retrieval with pseudo relevance feedback," in Proceedings of the 30th ACM International Conference on Information & Knowledge Management, pp. 3592–3596, 2021.

[26] R. Chandradevan, E. Yang, M. Yarmohammadi, and E. Agichtein, "Learning to enrich query representation with pseudo-relevance feedback for cross-lingual retrieval," in Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1790–1795, 2022.

[27] H. Li, S. Wang, S. Zhuang, A. Mourad, X. Ma, J. Lin, and G. Zuccon, "To interpolate or not to interpolate: Prf, dense and sparse retrievers," arXiv preprint arXiv:2205.00235, 2022.

[28] Y. Qu, Y. Ding, J. Liu, K. Liu, R. Ren, W. X. Zhao, D. Dong, H. Wu, and H. Wang, "Rocketqa: An optimized training approach to dense passage retrieval for open-domain question answering," in Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 5835–5847, 2021.

[29] G. Izacard and E. Grave, "Distilling knowledge from reader to retriever for question answering," in International Conference on Learning Representations, 2020.

[30] S. Hofstätter, S. Althammer, M. Schröder, M. Sertkan, and A. Hanbury, "Improving efficient neural ranking models with cross-architecture knowledge distillation," arXiv preprint arXiv:2010.02666, 2020.

[31] Y. Wang, J. Bai, Y. Wang, J. Zhang, W. Rong, Z. Ji, S. Wang, and J. Xiao, "Enhancing dual-encoders with question and answer cross-embeddings for answer retrieval," in Findings of the Association for Computational Linguistics: EMNLP 2021, pp. 2306–2315, 2021.

[32] M. Sung, J. Park, J. Kang, D. Chen, and J. Lee, "Optimizing test-time query representations for dense retrieval," in Findings of the Association for Computational Linguistics: ACL 2023 (A. Rogers, J. Boyd-Graber, and N. Okazaki, eds.), (Toronto, Canada), pp. 5731–5746, Association for Computational Linguistics, July 2023.

[33] D. Chicco, "Siamese neural networks: An overview," Artificial Neural Networks, pp. 73–94, 2021.

[34] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186, 2019.

[35] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with gpus," IEEE Transactions on Big Data, vol. 7, no. 3, pp. 535–547, 2019.

[36] G. Izacard, L. Hosseini, S. Riedel, P. Bojanowski, A. Joulin, and E. Grave, "Unsupervised dense information retrieval with contrastive learning," arXiv preprint arXiv:2112.09118, 2021.

[37] L. Xiong, C. Xiong, Y. Li, K.-F. Tang, J. Liu, P. N. Bennett, J. Ahmed, and A. Overwijk, "Approximate nearest neighbor negative contrastive learning for dense text retrieval," in International Conference on Learning Representations, 2020.

[38] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, and L. Deng, "Ms marco: A human generated machine reading comprehension dataset," in CoCo@ NIPs, 2016.

[39] R. Ren, Y. Qu, J. Liu, W. X. Zhao, Q. She, H. Wu, H. Wang, and J.-R. Wen, "Rocketqav2: A joint training method for dense passage retrieval and passage re-ranking," in Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 2825–2835, 2021.

[40] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, and I. Gurevych, "BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models," in Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2), 2021.

[41] Y. Qu, Y. Ding, J. Liu, K. Liu, R. Ren, W. X. Zhao, D. Dong, H. Wu, and H. Wang, "RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering," in Proceedings of the 2021 Conference of the North American Chapter of the Association for

Computational Linguistics: Human Language Technologies (K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tur, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, and Y. Zhou, eds.), (Online), pp. 5835–5847, Association for Computational Linguistics, June 2021.

[42] R. Ren, Y. Qu, J. Liu, W. X. Zhao, Q. She, H. Wu, H. Wang, and J.-R. Wen, "RocketQAv2: A joint training method for dense passage retrieval and passage re-ranking," in Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, eds.), (Online and Punta Cana, Dominican Republic), pp. 2825–2835, Association for Computational Linguistics, Nov. 2021.

[43] X. Zhang, X. Ma, P. Shi, and J. Lin, "Mr. tydi: A multi-lingual benchmark for dense retrieval," in Proceedings of the 1st Workshop on Multilingual Representation Learning, pp. 127–137, 2021.

[44] J. H. Clark, E. Choi, M. Collins, D. Garrette, T. Kwiatkowski, V. Nikolaev, and J. Palomaki, "Tydi qa: A benchmark for information-seeking question answering in typologically diverse languages," Transactions of the Association for Computational Linguistics, vol. 8, pp. 454–470, 2020.

[45] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, É. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, "Unsupervised cross-lingual representation learning at scale," in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 8440–8451, 2020.

[46] S. Longpre, Y. Lu, and J. Daiber, "Mkqa: A linguistically diverse benchmark for multilingual open domain question answering," Transactions of the Association for Computational Linguistics, vol. 9, pp. 1389–1406, 2021.

[47] J. Li, D. Li, C. Xiong, and S. Hoi, "Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation," in ICML, 2022.

[48] J. Xu, T. Mei, T. Yao, and Y. Rui, "Msr-vtt: A large video description dataset for bridging video and language," in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 5288–5296, 2016.

[49] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, et al., "Natural questions: a benchmark for question answering research," Transactions of the Association for Computational Linguistics, vol. 7, pp. 453–466, 2019.

[50] C. Sciavolino, Z. Zhong, J. Lee, and D. Chen, "Simple entity-centric questions challenge dense retrievers," in Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, (Online and Punta Cana, Dominican Republic), pp. 6138–6148, Association for Computational Linguistics, Nov. 2021.

[51] M. Fajcik, M. Docekal, K. Ondrej, and P. Smrz, "R2-D2: A modular baseline for open-domain question answering," in Findings of the Association for Computational Linguistics: EMNLP 2021, (Punta Cana, Dominican Republic), pp. 854–870, Association for Computational Linguistics, Nov. 2021.

[52] K. Santhanam, O. Khattab, J. Saad-Falcon, C. Potts, and M. Zaharia, "ColBERTv2: Effective and efficient retrieval via lightweight late interaction," in Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (M. Carpuat, M.-C. de Marneffe, and I. V. Meza Ruiz, eds.), (Seattle, United States), pp. 3715–3734, Association for Computational Linguistics, July 2022.